

Test Camera Code

```
using System;
using OpenTK;
using OpenTK.Graphics.OpenGL;
using OpenTK.Input;
using System.Drawing;
namespace OpenTK
{
    public class TestCamera : GameWindow
    {
        private Matrix4 cameraMatrix;
        private float[] mouseSpeed = new float[2];
        public TestCamera()
            : base(1024, 768)
        {
            GL.Enable(EnableCap.DepthTest);
        }

        #region ---OnLoad
        protected override void OnLoad(EventArgs e)//OnLoad
        {
            base.OnLoad(e);
            cameraMatrix = Matrix4.Translation(0f, -10f, 0f);
        }
        #endregion
        #region ---OnResize
        protected override void OnResize(EventArgs e)//OnResize Method
        {
            base.OnResize(e);
            GL.Viewport(ClientRectangle.X, ClientRectangle.Y, ClientRectangle.Width,
            ClientRectangle.Height);
            Matrix4 projection = Matrix4.CreatePerspectiveFieldOfView((float)Math.PI / 4,
            Width / (float)Height, 1.0f, 64.0f);
            GL.MatrixMode(MatrixMode.Projection);
            GL.LoadMatrix(ref projection);
        }
        #endregion
        #region ---OnRenderFrame
```

```

protected override void OnRenderFrame(FrameEventArgs e)//OnRenderFrame
{
    base.OnRenderFrame(e);
    GL.MatrixMode(MatrixMode.Modelview);
    GL.Clear(ClearBufferMask.ColorBufferBit | ClearBufferMask.DepthBufferBit);
    GL.LoadMatrix(ref cameraMatrix);
    //Display some Planes
    for (int x = -10; x <= 10; x++)
    {
        for (int z = -10; z <= 10; z++)
        {
            GL.PushMatrix();
            GL.Translate((float)x*5f,0f,(float)z*5f);
            GL.Begin(BeginMode.Quads);
            GL.Color3(Color.Red);//
            GL.Vertex3(1f,4f,0f);
            GL.Color3(Color.Blue);//
            GL.Vertex3(-1f,4f,0f);
            GL.Color3(Color.Brown);//
            GL.Vertex3(-1f,0f,0f);
            GL.Color3(Color.Maroon);//
            GL.Vertex3(1f,0f,0f);
            GL.End();
            GL.PopMatrix();
        }
    }
    SwapBuffers();
}
#endregion
#region ---OnUpdateFrame
protected override void OnUpdateFrame(FrameEventArgs e)// OnUpdateFrame
{
    base.OnUpdateFrame(e);
    if (Keyboard[Key.W])
    {
        cameraMatrix = Matrix4.Mult(cameraMatrix, Matrix4.Translation(0f, 0f, 10f
* (float)e.Time));
    }
    if (Keyboard[Key.S])
    {
        cameraMatrix = Matrix4.Mult(cameraMatrix, Matrix4.Translation(0f, 0f, -
10f * (float)e.Time));
    }
    if (Keyboard[Key.A])
    {
        cameraMatrix = Matrix4.Mult(cameraMatrix, Matrix4.Translation(10f *
(float)e.Time, 0f, 0f));
    }
    if (Keyboard[Key.D])
    {
        cameraMatrix = Matrix4.Mult(cameraMatrix, Matrix4.Translation(-10f *
(float)e.Time, 0f, 0f));
    }
}

```

```

    }
    if (Keyboard[Key.Q])
    {
        cameraMatrix = Matrix4.Mult(cameraMatrix, Matrix4.Translation(0f, 10f *
(float)e.Time, 0f));
    }
    if (Keyboard[Key.E])
    {
        cameraMatrix = Matrix4.Mult(cameraMatrix, Matrix4.Translation(0f, -10f *
(float)e.Time, 0f));
    }
    if (Keyboard[Key.Escape])
    {
        Exit();
        mouseSpeed[0] *= 0.2f;
        mouseSpeed[1] *= 0.2f;
        mouseSpeed[0] += Mouse.XDelta / 100f;
        mouseSpeed[1] += Mouse.YDelta / 100f;
        cameraMatrix = Matrix4.Mult(cameraMatrix, Matrix4.RotateY(mouseSpeed[0] *
(float)e.Time));
        cameraMatrix = Matrix4.Mult(cameraMatrix, Matrix4.RotateX(mouseSpeed[1] *
(float)e.Time));
    }
    #endregion
}
}

```

CUBE MOVE

```

using System;
using System.Threading;
using OpenTK;
using OpenTK.Graphics.OpenGL;
using OpenTK.Input;
using System.Drawing;
using OpenTK.Graphics;

namespace OpenTK
{
    class CubeMove : GameWindow
    {
        #region ---Fields
        const float rotation_speed = 90.0f;
        float angle;
        float x = 1.0f;
        float y = 0.0f;
        float z = 0.0f;
        float ZoomY = 5.0f;
        float ZoomZ = 5.0f;

```

```

float ZoomX = 0.0f;
#endregion

#region --- constructor ---
public CubeMove()
    : base(800, 600, new GraphicsMode(16, 16))
{ }
#endregion

#region ---OnLoad
protected override void OnLoad(EventArgs e)//OnLoad
{
    base.OnLoad(e);
    GL.ClearColor(Color.MidnightBlue);
    GL.Enable(EnableCap.DepthTest);
}
#endregion

#region ---OnResize
protected override void OnResize(EventArgs e)//OnResize Method
{
    base.OnResize(e);
    GL.Viewport(0, 0, Width, Height);
    double aspect_ratio = Width / (double)Height;
    OpenTK.Matrix4 perspective =
OpenTK.Matrix4.CreatePerspectiveFieldOfView(MathHelper.PiOver4, (float)aspect_ratio, 1,
640);

    //Matrix4 projection = Matrix4.CreatePerspectiveFieldOfView((float)Math.PI /
4, Width / (float)Height, 1.0f, 64.0f);
    GL.MatrixMode(MatrixMode.Projection);
    GL.LoadMatrix(ref perspective);
}
#endregion

#region --DrawCube
private void DrawCube()
{
    GL.Begin(BeginMode.Quads);

    GL.Color3(Color.Silver); //side 1
    GL.Vertex3(-1.0f, -1.0f, -1.0f);
    GL.Vertex3(-1.0f, 1.0f, -1.0f);
    GL.Vertex3(1.0f, 1.0f, -1.0f);
    GL.Vertex3(1.0f, -1.0f, -1.0f);
    //3915
    GL.Color3(Color.Honeydew); //side2
    GL.Vertex3(-1.0f, -1.0f, -1.0f);
    GL.Vertex3(1.0f, -1.0f, -1.0f);
    GL.Vertex3(1.0f, -1.0f, 1.0f);
    GL.Vertex3(-1.0f, 1.0f, -1.0f);
    //3797
    GL.Color3(Color.Moccasin); //side3
    GL.Vertex3(-1.0f, -1.0f, -1.0f);
    GL.Vertex3(-1.0f, -1.0f, 1.0f);
    GL.Vertex3(-1.0f, 1.0f, 1.0f);

```

```

        GL.Vertex3(-1.0f, 1.0f, -1.0f);
        //4005
        GL.Color3(Color.IndianRed); //side4
        GL.Vertex3(-1.0f, -1.0f, 1.0f);
        GL.Vertex3(1.0f, -1.0f, 1.0f);
        GL.Vertex3(1.0f, 1.0f, 1.0f);
        GL.Vertex3(-1.0f, 1.0f, 1.0f);
        //3204
        GL.Color3(Color.PaleVioletRed); //5
        GL.Vertex3(-1.0f, 1.0f, -1.0f);
        GL.Vertex3(-1.0f, 1.0f, 1.0f);
        GL.Vertex3(1.0f, 1.0f, 1.0f);
        GL.Vertex3(1.0f, 1.0f, -1.0f);
        //2817
        GL.Color3(Color.PaleVioletRed); //6
        GL.Vertex3(1.0f, -1.0f, -1.0f);
        GL.Vertex3(1.0f, 1.0f, -1.0f);
        GL.Vertex3(1.0f, 1.0f, 1.0f);
        GL.Vertex3(1.0f, -1.0f, 1.0f);
        //1602
        GL.End(); //For Commits select first CTRL+K+C
    }
    #endregion

    #region ---OnRenderFrame
    protected override void OnRenderFrame(FrameEventArgs e)
    {
        base.OnRenderFrame(e);
        GL.Clear(ClearBufferMask.ColorBufferBit | ClearBufferMask.DepthBufferBit);
        Matrix4 lookat = Matrix4.LookAt(ZoomX, ZoomY, ZoomZ, 0, 0, 0, 0, 1, 0);
        GL.MatrixMode(MatrixMode.Modelview);
        GL.LoadMatrix(ref lookat);
        angle += rotation_speed * (float)e.Time;
        GL.Rotate(angle, x, y, z);

        DrawCube();
        this.SwapBuffers(); // this is use for refreshing but it is necessary to put
        // at the end of code .if you put at the start it will filping
        Thread.Sleep(3);
    }
    #endregion

    #region ---OnUpdateFrame
    protected override void OnUpdateFrame(FrameEventArgs e)
    {
        base.OnUpdateFrame(e);
        if (Keyboard[Key.Escape])
        {
            this.Exit();
            return;
        }
        else if (Keyboard[Key.X])
        {
            x = 1.0f;
            y = 0.0f;
            z = 0.0f;
        }
        else if (Keyboard[Key.Y])
    }

```

```

        {
            x = 0.0f;
            y = 1.0f;
            z = 0.0f;
        }
        else if (Keyboard[Key.Z])
        {
            x = 0.0f;
            y = 0.0f;
            z = 1.0f;
        }
        else if (Keyboard[Key.W])
        {
            ZoomX++;
            ZoomY++;
            ZoomZ++;
        }
        else if (Keyboard[Key.S])
        {
            ZoomX--;
            ZoomY--;
            ZoomZ--;
        }
    }
    #endregion
}
}

```

ImageLoader

```

using System;
using System.Threading;
using OpenTK;
using OpenTK.Graphics.OpenGL;
using OpenTK.Input;
using System.Drawing;
using OpenTK.Graphics;
using System.Drawing.Imaging;

namespace OpenTK
{
    class ImageLoader : GameWindow
    {
        int texture;
        float x, y;
        #region ---Constructor---
        public ImageLoader()
        {
            this.Title = "OpenTK App 2 ";
            this.WindowBorder = WindowBorder.Fixed;
            this.ClientSize = new Size(800, 600);
        }
    }
    #endregion
}

```

```

#region ---OnLoad
protected override void OnLoad(EventArgs e)
{
    base.OnLoad(e);
    GL.ClearColor(Color.CornflowerBlue);
    GL.Ortho(0, 800, 600, 0, -1, 1);
    GL.Viewport(0, 0, 800, 600);
    GL.Enable(EnableCap.Texture2D);
    GL.Enable(EnableCap.Blend);
    GL.BlendFunc(BlendingFactorSrc.SrcAlpha,
BlendingFactorDest.OneMinusSrcAlpha);
    GL.Hint(HintTarget.PerspectiveCorrectionHint, HintMode.Nicest);
    GL.GenTextures(1, out this.texture);
    GL.BindTexture(TextureTarget.Texture2D, this.texture);
    Bitmap bitmap = new Bitmap("ship.png");
    bitmap.MakeTransparent(Color.Magenta);
    BitmapData data = bitmap.LockBits(new Rectangle(0, 0, bitmap.Width,
bitmap.Height),
    ImageLockMode.ReadOnly,
    System.Drawing.Imaging.PixelFormat.Format32bppArgb);
    GL TexImage2D(TextureTarget.Texture2D, 0,
        PixelInternalFormat.Rgba, data.Width, data.Height, 0,
OpenTK.Graphics.OpenGL.PixelFormat.Bgra, PixelType.UnsignedByte, data.Scan0);
    bitmap.UnlockBits(data);
    bitmap.Dispose();

    GL TexParameter(TextureTarget.Texture2D,
TextureParameterName.TextureMinFilter, (int)TextureMinFilter.Linear);
    GL TexParameter(TextureTarget.Texture2D,
        TextureParameterName.TextureMagFilter, (int)TextureMinFilter.Linear);
    this.x = 180f;
    this.y = 181.5f;
}
#endregion

#region ---OnRenderFrame
protected override void OnRenderFrame(FrameEventArgs e)
{
    base.OnRenderFrame(e);
    GL.Clear(ClearBufferMask.ColorBufferBit | ClearBufferMask.DepthBufferBit);
    GL.MatrixMode(MatrixMode.Projection);
    GL.LoadIdentity();
    GL.Begin(BeginMode.Quads);
    GL.TexCoord2(0, 0);
    GL.Vertex2(this.x - 40.0f, this.y - 41.5f);
    GL.TexCoord2(1, 0);
    GL.Vertex2(this.x + 40.0f, this.y - 41.5f);
    GL.TexCoord2(1, 1);
    GL.Vertex2(this.x + 40.0f, this.y + 41.5f);
    GL.TexCoord2(0, 1);
    GL.Vertex2(this.x - 40.0f, this.y + 41.5f);

    GL.End();
    GL.Flush();
    this.SwapBuffers();
}
}

```

```

#endregion
#region ---OnResize
protected override void OnResize(EventArgs e)
{
    base.OnResize(e);
}
#endregion
#region --- OnUpdateFrame
protected override void OnUpdateFrame(FrameEventArgs e)
{
    base.OnUpdateFrame(e);
    float dx = 0.0f;
    float dy = 0.0f;
    if (this.Keyboard[Key.Left])
        dx = -1.0f;
    else if(this.Keyboard[Key.Right])
        dx=1.0f;
    if (this.Keyboard[Key.Up])
        dy = -1.0f;
    else if(this.Keyboard[Key.Down])
        dy=1.0f;
    this.x+=100.0f*dx*(float)e.Time;
    this.y+=100.0f*dy*(float)e.Time;

}
#endregion
}

}

```